Ways to make new regular languages from known regular languages

Various modifications to languages that make another reg lang - this gives us a way to know some language is regular that doesn't rely on directly constructing an NFA/ reg exp / ~~reg~~ right-lin. grammar.

1) Given a language L (on an alphabet Σ), the language $\overline{L} = \{$all strings on Σ not in $L\}$ is also regular.

Pf: Since L is regular, we have a ~~NFA~~ DFA M that accepts L. If $M = (Q, \Sigma, \delta, q_0, F)$, then $\overline{M} = (Q, \Sigma, \delta, q_0, \boxed{Q \setminus F})$ accepts $\overline{L}$.

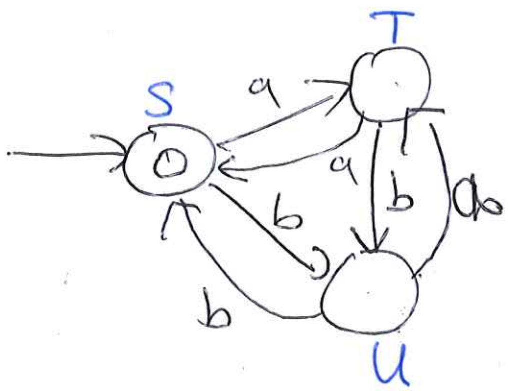
everything in Q not in F

2) Given ~~a~~ languages L_1 and L_2, $L_1 \cup L_2$ is regular.

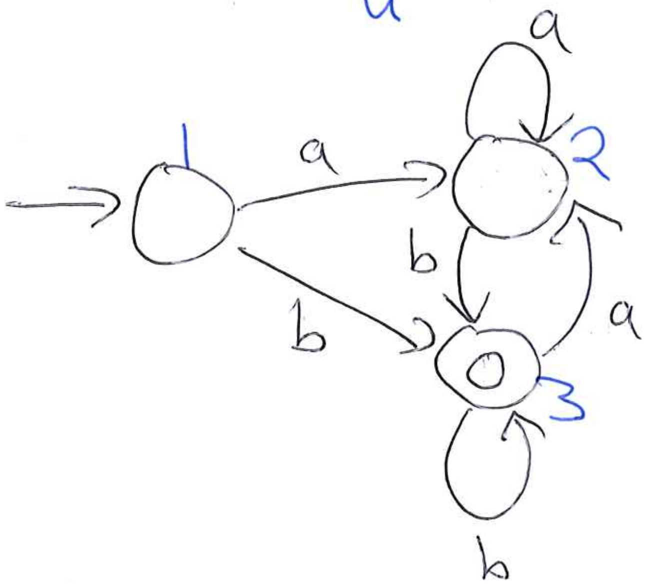
Pf: If we have ~~a~~ reg. exprs. e_1 + e_2 for L_1 + L_2, then $e_1 + e_2$ is a reg. expr for $L_1 \cup L_2$.

3) Given languages L_1 and L_2, $L_1 \cap L_2$ is regular.

Pf: Since L_1 + L_2 are regular, we have DFAs $M_1 = (Q_1, \Sigma, \delta_1, q_0^1, F_1)$ and $M_2 = (Q_2, \Sigma, \delta_2, q_0^2, F_2)$ for them.



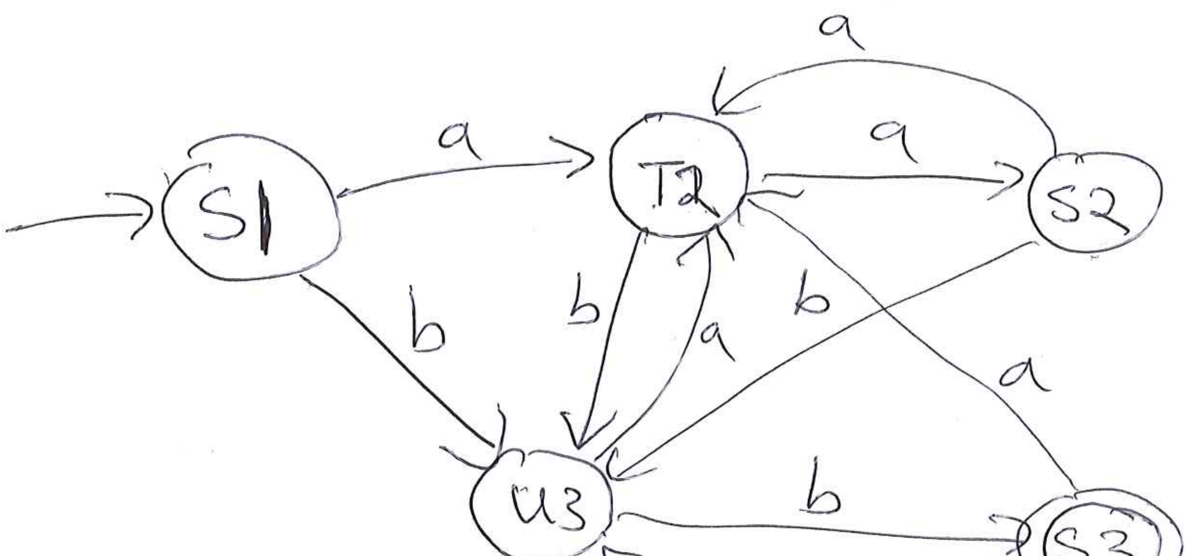
M_1



M_2

ababaa

New machine for $L_1 \cap L_2$:



Then the ~~[illegible]~~ DFA

$$M_1 \times M_2 = (\underbrace{Q_1 \times Q_2}, \Sigma, \delta, (q_0^1, q_0^2), F_1 \times F_2),$$

where

$$\delta((q_1, q_2), \ell) = (\delta_1(q_1, \ell), \delta_2(q_2, \ell)),$$

states in "$M_1 \times M_2$" look like (q_1, q_2), where $q_1 \in Q_1$, $q_2 \in Q_2$,

will accept precisely $L_1 \cap L_2$.

<u>Def</u>: A "homomorphism" from an ~~[illegible]~~ alphabet Σ to an alphabet T is a function from $\Sigma^* \longrightarrow T^*$ given by replacing every letter in a string on Σ^* by a fixed string in T^*.

E.g.: $\Sigma = \{a, b, c\}$

$\Gamma = \{0, 1, 2\}$

$h: \Sigma^* \longrightarrow \Gamma^*$ given by

replacing all a's with 01's,
all b's with 12's,
and all c's with λ's.

these could be other strings in a different h.

must account for every single letter of Σ.

So $h(abcbc) = 011212$.
$h(baabca) = 1201011201$

This h is a homomorphism.

Given a reg. lang. L and a homomorphism h, then

$h(L) = \{h(w) \mid w \in L\}$

is regular.

Pf: Given a reg. expr. e for L, h(e) is a reg. expr. for h(L).

Let L be a language on $\{a, b, c\}$.
Consider the language on $\{b, c\}$ given by all the strings that can be obtained by deleting all the a's from a string in L.
This is regular.

Pf: Take an NFA for L and replace all the a's with λ's.

Alternate Pf: ~~Use~~ consider the homomorphism h where
we replace a's with λ's
b's with b's
c's with c's

Then this lang. is $h(L)$.

Let L be a language on $\{a, b\}$,

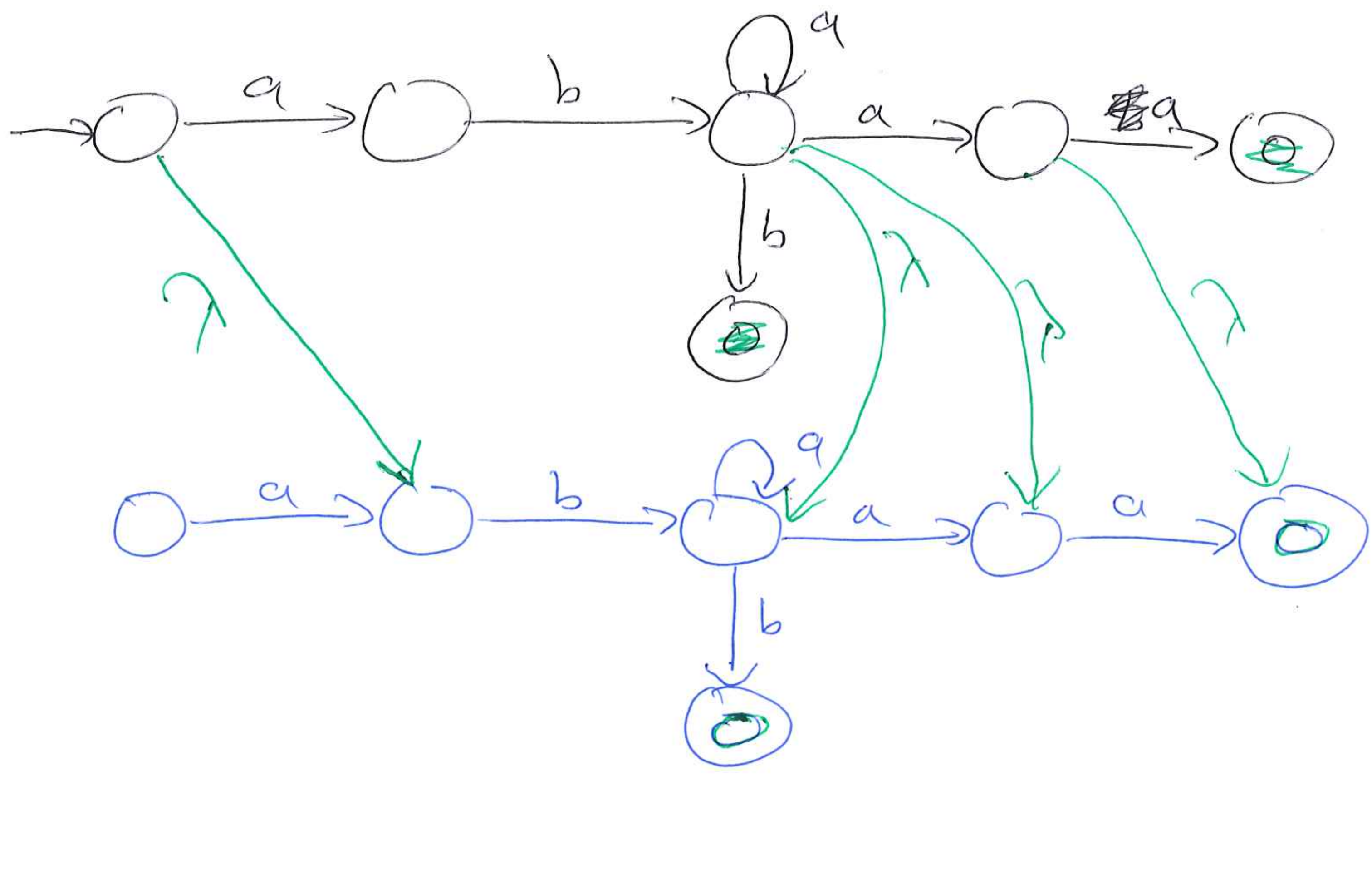
Consider $d(L) = \{$ all strings obtained by deleting a single a from a string in $L\}$

Then $d(L)$ is regular.

<u>Pf</u>: Given an NFA M for L.

---

example for thought:



modify M by

1) Make a second copy of M, called M'.

2) ~~Connec~~ for every ~~a~~ a-transition from q_1 to q_2 in M, make a λ transition from q_1 to copy-of-q_2.

in M in M'

3) The initial state is the init state in M; the final states are the final states in M' (but not those in M)